What is claimed is:

1.   1.   A compiler that optimizes a program to be compiled by changing the execution
2   order of instructions in the program, the compiler comprising:
3   an order constraint information obtaining unit that obtains order constraint
4   information indicating order constraints defined among a plurality of instructions in the
5   program, the order constraints defining the order in which the instructions should be
6   executed;
7   an order determination unit that sequentially determines the execution order for
8   each of the plurality of instructions based on the order constraint information;
9   a unit for analyzing the number of registers that analyzes the number of required
10   registers, which is the number of registers that will be required when the instructions
11   with its execution order determined among the plurality of instructions are executed;
12   an instruction detection unit that detects a combination of two instructions, in
13   which one instruction is a determined-order instruction for which the execution order
14   has been determined by the order determination unit, the other instruction is an
15   undetermined-order instruction for which the execution order has not been determined
16   by the order determination unit, and the order constraint information does not include
17   a constraint that the one instruction should be executed before the other instruction; and
18   an order determination reprocessing unit that, when the number of required
19   registers exceeds a predetermined number, changes the state of the one instruction
20   into the state in which the execution order has not been determined and causes the
21   order determination unit to determine the execution order so that the one instruction is
22   executed next to the other instruction.

1   2.   The compiler according to claim 1, wherein the instruction detection unit detects
2   an instruction that releases a register as the other instruction, and an instruction that
3   requires a new register allocated to it as the one instruction.

1   3.   The compiler according to claim 1, wherein the instruction detection unit detects

2   an instruction that releases a register as the other instruction, and an instruction to be

3   executed before a determined-order instruction that requires a new register allocated

4   to it as the one instruction, and the order determination reprocessing unit further

5   changes the state of all instructions to be executed after the determined-order

6   instruction in the order constraint information into the state in which the execution order

7   has not been determined.

1   4.   The compiler according to claim 1, wherein when a plurality of combinations of

2   the one instruction and the other instruction are detected by the instruction detection

3   unit, the order determination reprocessing unit selects from the plurality of

4   combinations a combination that minimizes the sum of the depth of order constraint

5   from a start point of the program to the other instruction and the depth of order

6   constraint from the one instruction to an end point of the program, and the order

7   determination reprocessing unit causes the order determination unit to determine the

8   execution order using the other instruction and the one instruction included in the

9   selected combination.

1   5.   The compiler according to claim 1, wherein when the number of required

2   registers exceeds the predetermined number, the order determination reprocessing

3   unit adds an order constraint that the determined-order instruction should be executed

4   next to the undetermined-order instruction to the order constraint information, and

5   thereby causes the order determination unit to determine the execution order so that

6   the determined-order instruction is executed next to the undetermined-order instruction.

1   6.   The compiler according to claim 5, wherein the order constraint information

2   obtaining unit obtains, as the order constraint information, an order constraint graph

3   that represents each instruction in the program as a node and order constraints under

4   which a plurality of instructions should be executed as directed edges, the order

5   determination unit determines the execution order based on the order constraint graph

6   so that an instruction represented by a start node of a directed edge is executed

7   before an instruction represented by an end node of the directed edge,

8   the instruction detection unit detects that the order constraint information does not

9   include an order constraint that the one instruction should be executed before the other

10   instruction by detecting a combination of two instructions in which a node representing

11   the one instruction cannot reach a node representing the other instruction on the order

12   constraint graph, and the order determination reprocessing unit adds an order

13   constraint that the other instruction should be executed next to the one instruction to the

14   order constraint information by generating a directed edge from the node representing

15   the undetermined-order instruction to the node representing the other instruction.

1   7.   A compiler program for causing a computer to function as a compiler that

2   optimizes a program to be compiled by changing the execution order of instructions

3   in the program, wherein the compiler program causes the computer to function as:

4   an order constraint information obtaining unit that obtains order constraint

5   information indicating order constraints defined among a plurality of instructions in the

6   program, the order constraints defining the order in which the instructions should be

7   executed;

8   an order determination unit that sequentially determines the execution order for

9   each of the plurality of instructions based on the order constraint information;

10   a unit for analyzing the number of registers that analyzes the number of required

11   registers, which is the number of registers that will be required when the instructions

12   with its execution order determined among the plurality of instructions are executed;

13   an instruction detection unit that detects a combination of two instructions, in

14   which one instruction is a determined-order instruction for which the execution order

15   has been determined by the order determination unit, the other instruction is an

16   undetermined-order instruction for which the execution order has not been determined

17   by the order determination unit, and the order constraint information does not include

18 a constraint that the one instruction should be executed before the other instruction; and

19 an order determination reprocessing unit that, when the number of required

20 registers exceeds a predetermined number, changes the state of the one instruction

21 into the state in which the execution order has not been determined and causes the

22 order determination unit to determine the execution order so that the one instruction is

23 executed next to the other instruction.

1 8. The compiler program according to claim 7, wherein the instruction detection

2 unit detects an instruction that releases a register as the other instruction, and an

3 instruction that requires a new register allocated to it as the one instruction.

1 9. The compiler program according to claim 7, wherein when the number of

2 required registers exceeds the predetermined number, the order determination

3 reprocessing unit adds an order constraint that the one instruction should be executed

4 next to the other instruction to the order constraint information, and thereby causes the

5 order determination unit to determine the execution order so that the one instruction is

6 executed next to the other instruction.

1 10. A recording medium with the compiler program according to claim 7 recorded

2 on it.

1 11. A compiling method for optimizing a program to be compiled by changing the

2 execution order of instructions in the program with a computer, the method comprising:

3 obtaining order constraint information indicating order constraints defined

4 among a plurality of instructions in the program, the order constraints defining the

5 order in which the instructions should be executed;

6 sequentially determining the execution order for each of the plurality of

7 instructions based on the order constraint information;

8

9   analyzing the number of required registers, which is the number of registers that

10   will be required when the instructions with its execution order determined among the

11   plurality of instructions are executed;

12   detecting a combination of two instructions, in which one instruction is a

13   determined-order instruction for which the execution order has been determined by the

14   order determination unit, the other instruction is an undetermined-order instruction for

15   which the execution order has not been determined by the order determination unit,

16   and the order constraint information does not include a constraint that the one

17   instruction should be executed before the other instruction; and

18   when the number of required registers exceeds a predetermined number,

19   changing the state of the one instruction into the state in which the execution order has

20   not been determined and causing the order determination unit to determine the

21   execution order so that the one instruction is executed next to the other instruction.